AUTOMATIC LOG-IN/LOG-OUT COMPUTER SYSTEM

Patent Number: JP9251331

Publication date: 1997-09-22

Inventor(s): SASAKI MASATOMI

Applicant(s): HITACHI LTD

Requested Patent: JP9251331

Application Number: JP19960060588 19960318

Priority Number(s):

IPC Classification: G06F1/26; G06F15/00; G06K19/00

EC Classification:

Equivalents:

Abstract

PROBLEM TO BE SOLVED: To reduce a trouble for saving the power of a system and for security protection by executing log-in and log-out at the same time as power on/off by inserting/extracting an ID card and the other cards. SOLUTION: When the ID card 30 is inserted into the ID card insertion port 21 of a card reader 20, a power switch 22 is turned on and a computer system 10 is started. Individual information of the ID card 30 is read and ID is judged whether the content is previously registered in a computer system main body 11 or not, When ID is not registered, power is turned off. When it is registered, log-in is executed and an operator can input data from a keyboard 13 and he can execute work while he views a display device 12. For terminating of interrupting work. the power switch 22 senses the extraction of the ID card 30 when it is extracted, the computer system 10 saves a file during editing and power is turned off after a log-out processing.

(19)日本国特許庁（ＪＰ）　(12)**公開特許公報**（Ａ）　(11)特許出願公開番号

特開平9－251331

(43)公開日　平成９年(1997)９月22日

(51)Int.Cl.[6]		識別記号	庁内整理番号	ＦＩ		技術表示箇所
Ｇ０６Ｆ	1/26			Ｇ０６Ｆ	1/00	３３４Ａ
	15/00	３３０			15/00	３３０Ｂ
Ｇ０６Ｋ	19/00			Ｇ０６Ｋ	19/00	Ｔ

審査請求　未請求　請求項の数１　ＯＬ　（全３頁）

(21)出願番号　特願平8－60588

(22)出願日　平成８年(1996)３月18日

(71)出願人　000005108
株式会社日立製作所
東京都千代田区神田駿河台四丁目６番地

(72)発明者　佐々木　正富
愛知県尾張旭市晴丘町池上１番地株式会社日立製作所オフィスシステム事業部内

(74)代理人　弁理士　小川　勝男

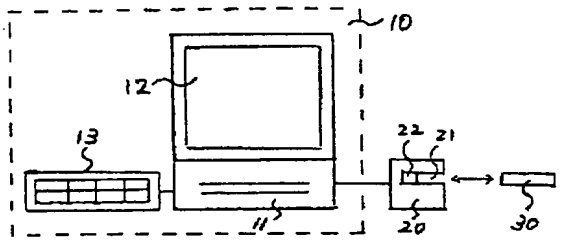
(54)【発明の名称】　自動ログイン・ログアウト計算機システム

(57)【要約】

【課題】計算機システムの省電力化とセキュリティ保護に対する、オペレータの電源ＯＮ・ＯＦＦ並びにログイン・ログアウトの手間を軽減する。

【解決手段】図１のように、計算機システム１０に電源スイッチ２２を内蔵したカードリーダ２０を取り付ける。これにより、ＩＤカード３０の抜き差しによって計算機システム１０の電源をＯＮ・ＯＦＦすると共に個人情報を読み込み、オペレータの自動ログイン・ログアウトが可能となる。

図1



1

【特許請求の範囲】

【請求項１】ＩＤカードおよびその他カード類の抜き差しによって、電源をＯＮ・ＯＦＦすると同時に、ログイン・ログアウトを自動的に行なえる計算機システム。

【発明の詳細な説明】

【０００１】

【発明の属する技術分野】本発明は、計算機システムの省電力化と、オペレータ操作の手間軽減に関する。

【０００２】

【従来の技術】従来の技術は、ログイン・ログアウトをオペレータがキーボード入力することにより行なうことが一般的である。また、特開昭６０－２２５９６４号に記載のように、オペレータはログイン前に計算機システムの電源をあらかじめＯＮにしておく必要があった。

【０００３】

【発明が解決しようとする課題】前記従来技術では、オペレータは席を外す度にセキュリティ保護のため、計算機システムのログイン・ログアウト操作をキーボード入力により行なっており、手間がかかった。また、ログインする前の計算機システムの電源ＯＮや、ログアウトした後の省電力のための電源ＯＦＦの手間が必要であり、問題があった。

【０００４】本発明は、上記のような問題を解決するために考案したものであって、その目的は計算機システムの省電力化を図るとともに、セキュリティ保護に対するオペレータの手間を軽減することである。

【０００５】

【課題を解決するための手段】上記目的を達成するために、計算機システムにカードリーダを取り付け、カードリーダにはＩＤカードの抜き差しによって計算機システムの電源をＯＮ・ＯＦＦするスイッチを取り付ける。

【０００６】

【発明の実施の形態】以下、本発明の一実施例を図１、図２により説明する。図１は、本発明の特徴を有するシステム構成図を示す。図２は、図１のシステムの動作を示す流れ図である。

【０００７】図１において、１０は計算機システム、１１は計算機システム本体、１２は表示装置、１３はキーボード、２０はカードリーダ、２１はＩＤカード挿入口、２２は電源スイッチ、３０はＩＤカードである。

2

【０００８】次に、図２の流れ図に基づいて、図１のシステムの動作を説明する。

【０００９】まず、図１に示すＩＤカード３０をカードリーダ２０のＩＤカード挿入口２１に挿入すると（ステップ１００）、電源スイッチ２２がＯＮされ（ステップ１０２）、計算機システム１０が起動する（ステップ１０４）。続いて、ＩＤカード３０の個人情報がカードリーダ２０によって読み込まれ、その内容があらかじめ計算機システム本体１１に登録されているＩＤか否かを判定し（ステップ１０６）、登録されていないＩＤであれば電源ＯＦＦとなり（ステップ１１８）、登録されていればログインされ（ステップ１０８）、オペレータはキーボード１３からの入力が可能となり、表示装置１２を見ながら作業が行なえる（ステップ１１０）。次に、オペレータが作業を終了、又は中断する場合には、ＩＤカード３０を引き抜くと（ステップ１１２）、電源スイッチ２２がＩＤカード３０が引き抜かれたことを感知し、計算機システム１０に作業終了の信号を送り、それを受けた計算機システム１０は、自動的にオペレータ編集中のファイル等をセーブし（ステップ１１４）、ログアウト処理を行なった後（ステップ１１６）、電源をＯＦＦする（ステップ１１８）。

【００１０】

【発明の効果】本発明により、計算機システムの省電力化とセキュリティ保護に対する、オペレータの電源ＯＮ・ＯＦＦ並びにログイン・ログアウトの手間を軽減することができる。

【図面の簡単な説明】

【図１】本発明のシステム構成図である。

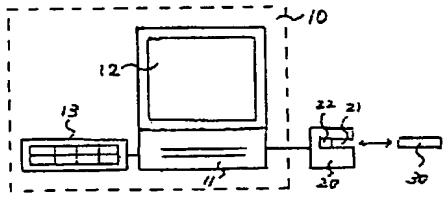
【図２】本システムの動作を示す流れ図である。

【符号の説明】

１０…計算機システム、１１…計算機システム本体、１２…表示装置、１３…キーボード、　２０…カードリーダ、　２１…ＩＤカード挿入口、２２…電源スイッチ、　３０…ＩＤカード。

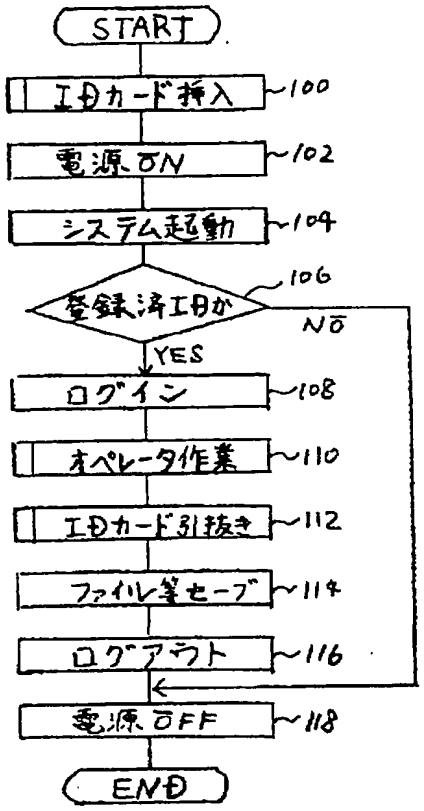
【図１】

図1



【図2】

図2.



図中 [ ] はオペレータが行ない、[ ] はシステムが自動的に行なう。

DEVICE AND METHOD FOR MANAGING NETWORK

Patent Number: JP2000353142
Publication date: 2000-12-19
Inventor(s): HAMADA NOBORU
Applicant(s): CANON INC
Requested Patent: JP2000353142
Application Number: JP19990165570 19990611
Priority Number(s):
IPC Classification: G06F13/00; G06F13/14
EC Classification:
Equivalents:

Abstract

PROBLEM TO BE SOLVED: To reduce traffic for searching a network device.
SOLUTION: In the case of searching a network device, a packet for search is executed once at the time of starting the search (S1301), the research of each device searched at the time of starting is repeated one by one at a prescribed interval (S1305). Consequently a situation where network traffic increases can be prevented by broadcasting the packet for search. Since the broadcast packet is periodically transmitted, the broadcast packet is successively transmitted to devices registered in a device table instead of updating the device table and only whether each device responds to the packet or not can be checked. Consequently the device table can be updated in a comparatively small network traffic.

(19)日本国特許庁（ＪＰ）　(12)公開特許公報（Ａ）　(11)特許出願公開番号
特開2000－353142
（P2000－353142A）
(43)公開日　平成12年12月19日(2000.12.19)

(51)Int.Cl.'		識別記号	FI		テーマコード'(参考)
Ｇ０６Ｆ	13/00	３５７	Ｇ０６Ｆ 13/00	３５７Ａ	５Ｂ０１４
		３５３		３５３Ｂ	５Ｂ０８９
	13/14	３３０	13/14	３３０Ａ	

審査請求　未請求　請求項の数６　ＯＬ　（全 20 頁）

(21)出願番号　特願平11－165570

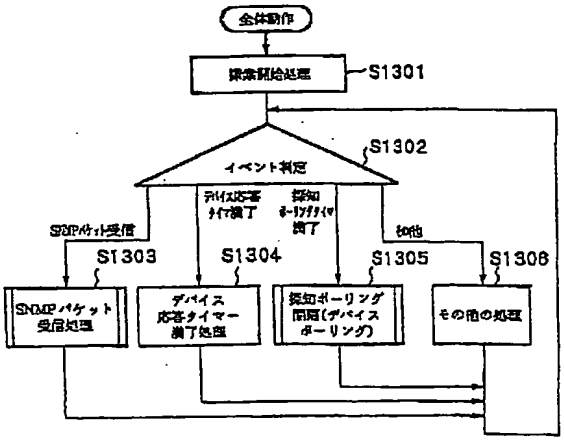
(22)出願日　平成11年６月11日(1999.6.11)

(71)出願人　000001007
キヤノン株式会社
東京都大田区下丸子３丁目30番２号
(72)発明者　浜田　昇
東京都大田区下丸子３丁目30番２号　キヤノン株式会社内
(74)代理人　100076428
弁理士　大塚　康徳　（外２名）

Ｆターム(参考)　5B014 FB04 HC02
5B089 GB02 HA06 JA35 JB15 KA13
KB04 KC20 KC30 KC44

(54)【発明の名称】　ネットワーク管理装置及び方法

(57)【要約】
【課題】ネットワークデバイスの探索のためのトラフィックを減らす。
【解決手段】ネットワークデバイスの探索時には、起動時に探知のためのパケットを一度行った後（Ｓ１３０１）、所定の間隔で、起動時に探知したデバイスに対して、一度に一つずつ再探知を繰り返す（Ｓ１３０５）。こうして、探知のためにパケットをブロードキャストしてネットワークトラフィックをあげてしまう事態を防止する。



【特許請求の範囲】

【請求項１】　ブロードキャストパケットによってネットワークに接続されているデバイスを検出するためのネットワークデバイス探知手段と、

前記ネットワークデバイス探知手段により検出されたデバイスを登録するためのデバイステーブルと、

前記デバイステーブルに登録されたデバイスに対して、所定間隔で順次探知シーケンスを遂行する個別デバイス探知手段と、

前記個別デバイス探知手段により検出された情報により、前記デバイステーブルを更新するデバイステーブル更新手段とを備えることを特徴とするネットワークデバイス管理装置。

【請求項２】　前記デバイステーブルの登録内容の出力を指示された際に、前記デバイステーブルに登録された各デバイスに対して探知シーケンスを遂行し、検出された情報により前記デバイステーブルを更新する手段と、前記デバイステーブルを出力する出力手段とを更に備えることを特徴とするネットワークデバイス管理装置。

【請求項３】　ブロードキャストパケットによってネットワークに接続されているデバイスを検出し、検出されたデバイスをデバイステーブルに登録するためのネットワークデバイス探知工程と、

前記デバイステーブルに登録されたデバイスに対して、所定間隔で順次探知シーケンスを遂行する個別デバイス探知工程と、

前記個別デバイス探知工程により検出された情報により、前記デバイステーブルを更新するデバイステーブル更新工程とを備えることを特徴とするネットワークデバイス管理方法。

【請求項４】　前記デバイステーブルの登録内容の出力を指示された際に、前記デバイステーブルに登録された各デバイスに対して探知シーケンスを遂行し、検出された情報により前記デバイステーブルを更新する工程と、前記デバイステーブルを出力する出力工程とを更に備えることを特徴とするネットワークデバイス管理方法。

【請求項５】　コンピュータにより、

ブロードキャストパケットによってネットワークに接続されているデバイスを検出するためのネットワークデバイス探知手段と、

前記ネットワークデバイス探知手段により検出されたデバイスを登録するためのデバイステーブルと、

前記デバイステーブルに登録されたデバイスに対して、所定間隔で順次探知シーケンスを遂行する個別デバイス探知手段と、

前記個別デバイス探知手段により検出された情報により、前記デバイステーブルを更新するデバイステーブル更新手段とを実現するためのコンピュータプログラムを格納するコンピュータ可読メモリ。

【請求項６】　前記デバイステーブルの登録内容の出力を指示された際に、前記デバイステーブルに登録された各デバイスに対して探知シーケンスを遂行し、検出された情報により前記デバイステーブルを更新する手段と、前記デバイステーブルを出力する出力手段とを更に実現するためのコンピュータプログラムを格納する請求項５に記載のコンピュータ可読メモリ。

【発明の詳細な説明】

【０００１】

【発明の属する技術分野】本発明は、例えばコンピュータネットワークに接続されたデバイス等を管理するためのネットワークデバイス管理装置及び方法に関する。

【０００２】

【従来の技術】近年、コンピュータを相互に接続したローカルエリアネットワーク（ＬＡＮ）が普及しており、このようなローカルエリアネットワークは、ビルのフロアまたはビル全体、ビル群（構内）、地域、あるいはさらに大きいエリアにわたって構築することができる。このようなネットワークは更に相互接続され、世界的規模のネットワークにも接続することができる。このような相互接続されたそれぞれのＬＡＮは、多様なハードウェア相互接続技術といくつものネットワークプロトコルを持つことがある。

【０００３】他と切り離された簡単なＬＡＮは個々のユーザが管理することができる。すなわち、ユーザが機器を取り替えたり、ソフトウェアをインストールしたり、問題点を診断したりすることができる。

【０００４】しかし一方、規模の大きい複雑なＬＡＮや相互接続された大きなＬＡＮグループは「管理」を必要とする。「管理」とは、人間のネットワーク管理者とその管理者が使用するソフトウェアの両方による管理を意味する。本願においては、「管理」とはシステム全体を管理するためのソフトウェアによる管理を意味し、「ユーザ」とはネットワーク管理ソフトウェアを使用する人を意味するものとする。このユーザは、通常、システム管理責任者である。ユーザは、ネットワーク管理ソフトウェアを使うことによって、ネットワーク上で管理データを得て、このデータを変更することができる。

【０００５】大規模ネットワークシステムは、通常、機器の増設と除去、ソフトウェアの更新、及び問題の検出などを絶えず行うことが必要な動的システムである。一般に、様々な人が所有する、様々な業者から供給される様々なシステムが存在する。

【０００６】このような大規模ネットワークシステムを構成するネットワーク上のデバイスを管理するための方法として、これまでにいくつかの試みが数多くの標準機関でなされている。国際標準化機構（ＩＳＯ）は開放型システム間相互接続（Open System Interconnection, ＯＳＩ）モデルと呼ばれる汎用基準フレームワークを提供した。ネットワーク管理プロトコルのＯＳＩモデルは、共通管理情報プロトコル（Comon Management Infor

mation Protocol，ＣＭＩＰ）と呼ばれる。ＣＭＩＰはヨーロッパの共通ネットワーク管理プロトコルである。

【０００７】また近年では、より共通性の高いネットワーク管理プロトコルとして、簡易ネットワーク管理プロトコル（Simple Network Management Protocol，ＳＮＭＰ）と呼ばれるＣＭＩＰに関連する一変種のプロトコルがある。（「ＴＣＰ／ＩＰネットワーク管理入門実用的な管理をめざして」Ｍ．Ｔ．ローズ＝著／西田竹志＝訳（株）トッパン発行１９９２年８月２０日初版を参照。）このＳＮＭＰネットワーク管理技術によれば、ネットワーク管理システムには、少なくとも１つのネットワーク管理ステーション（ＮＭＳ）、各々がエージェントを含むいくつかの管理対象ノード、及び管理ステーションやエージェントが管理情報を交換するために使用するネットワーク管理プロトコルが含まれる。ユーザは、ＮＭＳ上でネットワーク管理ソフトウェアを用いて管理対象ノード上のエージェントソフトウェアと通信することにより、ネットワーク上のデータを得、またデータを変更することができる。

【０００８】ここでエージェントとは、各々のターゲット装置についてのバックラウンドプロセスとして走るソフトウェアである。ユーザがネットワーク上の装置に対して管理データを要求すると、管理ソフトウェアはオブジェクト識別情報を管理パケットまたはフレームに入れてターゲットエージェントへ送り出す。エージェントは、そのオブジェクト識別情報を解釈して、そのオブジェクト識別情報に対応するデータを取り出し、そのデータをパケットに入れてユーザに送り返す。時には、データを取り出すために対応するプロセスが呼び出される場合もある。

【０００９】またエージェントは、自分の状態に関するデータをデータベースの形式で保持している。このデータベースのことを、ＭＩＢ(Management Information Base)と呼ぶ。図４は、ＭＩＢの構造を示す概念図である。図４に示すように、ＭＩＢは木構造のデータ構造をしており、全てのノードが一意に番号付けされている。図４において、かっこ内に書かれている番号が、そのノードの識別子である。例えば、図４においてノード４０１の識別子は１である。ノード４０２の識別子は、ノード４０１の下の３なので、１・３と表記される。同様にして、ノード４０３の識別子は、１・３・６・１・２と表記される。このノードの識別子のことを、オブジェクト識別子(OBJECT IDENTIFIER)と呼ぶ。

【００１０】このＭＩＢの構造は、管理情報構造（ＳＭＩ：Structure of Management Information）と呼ばれ、RFC1155 Structure and Identification of Management Information for TCP/IP-based Internetsで規定されている。

【００１１】図４には、標準として規定されているＭＩＢのうち、一部のもののみを抜き出して記載してある。

【００１２】次に、ＳＮＭＰプロトコルについて簡単に説明する。ネットワーク管理ユーティリティソフトウェアが動作しているＰＣ（以下、マネージャと呼称する）とＳＮＭＰエージェントが動作している管理対象ネットワークデバイス（以下、エージェントと呼称する）とは、ＳＮＭＰプロトコルを用いて通信を行う。ＳＮＭＰプロトコルには５種類のコマンドがあり、それぞれＧｅｔ－ｒｅｑｕｅｓｔ、Ｇｅｔ－ｎｅｘｔ－ｒｅｑｕｅｓｔ、Ｇｅｔ－ｒｅｓｐｏｎｓｅ、Ｓｅｔ－ｒｅｑｕｅｓｔ、Ｔｒａｐと呼ばれる。これらのコマンドがマネージャとエージェントの間でやりとりされる様子を図８に示す。

【００１３】Ｇｅｔ－ｒｅｑｕｅｓｔおよびＧｅｔ－ｎｅｘｔ－ｒｅｑｕｅｓｔは、マネージャがエージェントのＭＩＢオブジェクトの値を取得するために、マネージャがエージェントに対して送出するコマンドである。このコマンドを受け取ったエージェントは、ＭＩＢの値をマネージャに通知するために、マネージャに対してＧｅｔ－ｒｅｓｐｏｎｓｅコマンドを送出する（８０１および８０２）。

【００１４】Ｓｅｔ－ｒｅｑｕｅｓｔは、マネージャがエージェントのＭＩＢオブジェクトの値を設定するために、マネージャがエージェントに対して送出するコマンドである。このコマンドを受け取ったエージェントは、設定結果をマネージャに通知するために、マネージャに対してＧｅｔ－ｒｅｓｐｏｎｓｅコマンドを送出する（８０３）。

【００１５】Ｔｒａｐは、エージェントが自分自身の状態の変化をマネージャに対して通知するために、エージェントがマネージャに対して送出するコマンドである（８０４）。

【００１６】図７に、Ｔｒａｐ以外のコマンド、即ちＧｅｔ－ｒｅｑｕｅｓｔ、Ｇｅｔ－ｎｅｘｔ－ｒｅｑｕｅｓｔ、Ｇｅｔ－ｒｅｓｐｏｎｓｅおよびＳｅｔ－ｒｅｑｕｅｓｔのフォーマットを示す。

【００１７】７００は、ＳＮＭＰメッセージを示す。ＳＮＭＰメッセージは、バージョン７０１、コミュニティ名７０２およびＰＤＵと呼ばれる領域７０３からなる。ＰＤＵ７０３を詳細に示したものが７１０である。ＰＤＵ７１０は、ＰＤＵタイプ７１１、リクエストＩＤ７１２、エラーステータス７１３、エラーインデックス７１４およびＭＩＢ情報７１５からなる。ＰＤＵタイプ７１１には、コマンドを識別する値が格納される。即ちこのフィールドの値が０であるならばＧｅｔ－ｒｅｑｕｅｓｔ、１であるならばＧｅｔ－ｎｅｘｔ－ｒｅｑｕｅｓｔ、２であるならばＧｅｔ－ｒｅｓｐｏｎｓｅ、３であるならばＳｅｔ－ｒｅｑｕｅｓｔと識別される。また、エラーステータス７１３には、エラー情報を示す値が格納される。エラーが無い場合には、このフィールドの値は０である。また、ＭＩＢ情報７１５には、オブジェク

トＩＤとその値が組みになって格納される。

【００１８】次に、管理が必要な大規模なネットワークについて説明する。

【００１９】図１は、プリンタをネットワークに接続するためのネットワークボード（ＮＢ）１０１を、開放型アーキテクチャを持つプリンター１０２へつなげた場合を示す図である。ＮＢ１０１はローカルエリアネットワーク（ＬＡＮ）１００へ、例えば、同軸コネクタをもつEthernetインターフェース10Base-2や、ＲＪ－４５を持つ10Base-T等のＬＡＮインターフェースを介してつながれている。

【００２０】ＰＣ１０３やＰＣ１０４等の複数のパーソナルコンピューター（ＰＣ）もまた、１００に接続されており、ネットワークオペレーティングシステムの制御の下、これらのＰＣはＮＢ１０１と通信することができる。ＰＣの一つ、例えばＰＣ１０３を、ネットワーク管理部として使用するように指定することができる。ＰＣに、ＰＣ１０４に接続されているプリンター１０５のようなプリンターを接続してもよい。

【００２１】また、ＬＡＮ１００にファイルサーバー１０６が接続されており、これは大容量（例えば１００億バイト）のネットワークディスク１０７に記憶されたファイルへのアクセスを管理する。プリントサーバー１０８は、接続されたプリンター１０９ａ及び１０９ｂ、又は遠隔地にあるプリンター１０５などのプリンターに印刷を行わせる。また他の図示しない周辺機器をＬＡＮ１００に接続してもよい。

【００２２】更に詳しくは、図１に示すネットワークは、様々なネットワークメンバー間で効率良く通信を行うために、NovellやUNIXのソフトウェアなどのネットワークソフトウェアを使用することができる。どのネットワークソフトウェアを使用することも可能であるが、例えば、Novell社のNetWare（Novell社の商標。以下省略）ソフトウェアを使用することができる。このソフトウェアパッケージに関する詳細な説明は、NetWareパッケージに同梱されているオンラインドキュメンテーションを参照のこと。これは、Novell社からNetWareパッケージとともに購入可能である。

【００２３】図１の構成について簡潔に説明すると、ファイルサーバー１０６は、ＬＡＮメンバー間でデータのファイルの受信や、記憶、キューイング、キャッシング、及び送信を行う、ファイル管理部としての役割を果たす。例えば、ＰＣ１０３及びＰＣ１０４それぞれによって作られたデータファイルは、ファイルサーバー１０６へ送られ、ファイルサーバー１０６はこれらのデータファイルを順に並べ、そしてプリントサーバー１０８からのコマンドに従って、並べられたデータファイルをプリンター１０９ａへ送信する。

【００２４】またＰＣ１０３とＰＣ１０４はそれぞれ、データファイルの生成や、生成したデータファイルのＬＡＮ１００への送信や、また、ＬＡＮ１００からのファイルの受信や、更にそのようなファイルの表示及び／又は処理を行うことのできる、通常のＰＣで構成される。図１にパーソナルコンピューター機器が示されているが、ネットワークソフトウェアを実行するのに適切であるような、他のコンピューター機器を含んでもよい。例えば、UNIXのソフトウェアを使用している場合に、UNIXワークステーションをネットワークに含んでもよく、これらのワークステーションは、適切な状況下で、図示されているＰＣと共に使用することができる。

【００２５】通常、ＬＡＮ１００などのＬＡＮは、一つの建物内の一つの階又は連続した複数の階でのユーザーグループ等の、幾分ローカルなユーザーグループにサービスを提供する。例えば、ユーザーが他の建物や他県に居るなど、あるユーザーが他のユーザーから離れるに従って、ワイドエリアネットワーク（ＷＡＮ）を作ってもよい。ＷＡＮは、基本的には、いくつかのＬＡＮを高速度サービス総合デジタルネットワーク（ＩＳＤＮ）電話線等の高速度デジタルラインで接続して形成された集合体である。従って、図１に示すように、ＬＡＮ１００と、ＬＡＮ１１０と、ＬＡＮ１２０とは変調／復調（ＭＯＤＥＭ）／トランスポンダー１３０及びバックボーン１４０を介して接続されＷＡＮを形成する。これらの接続は、数本のバスによる単純な電気的接続である。それぞれのＬＡＮは専用のＰＣを含み、また、必ずしも必要なわけではないが、通常はファイルサーバー及びプリントサーバーを含む。

【００２６】従って図１に示すように、ＬＡＮ１１０は、ＰＣ１１１と、ＰＣ１１２と、ファイルサーバー１１３と、ネットワークディスク１１４と、プリントサーバー１１５と、プリンター１１６及びプリンター１１７とを含む。対照的に、ＬＡＮ１２０はＰＣ１２１とＰＣ１２２のみを含む。ＬＡＮ１００と、ＬＡＮ１１０と、ＬＡＮ１２０とに接続されている機器は、ＷＡＮ接続を介して、他のＬＡＮの機器の機能にアクセスすることができる。

【００２７】エージェントの実装例として、プリンタをネットワークに接続するためのネットワークボード上にエージェントを実装することが考えられる。これにより、プリンタをネットワーク管理ソフトウェアによる管理の対象とすることができる。ユーザは、ネットワーク管理ソフトウェアを用いて制御対象のプリンタの情報を得、また状態を変更することができる。より具体的には、例えばプリンタの液晶ディスプレイに表示されている文字列を取得したり、デフォルトの給紙カセットを変更したりすることができる。以下、エージェントを実装したネットワークボード（ＮＢ）をプリンタに接続する実施形態について説明する。図２に示すように、好ましくは、ＮＢ１０１は、プリンター１０２の内部拡張Ｉ／Ｏスロットに内蔵されており、ＮＢ１０１は、下に示す

処理及びデータ記憶機能を持つ「埋め込まれた」ネットワークノードとなる。このＮＢ１０１の構成により、大きなマルチエリアＷＡＮネットワークを統括及び管理するための、特徴的な補助機能を持つという利点をもたらす。これらの補助機能は、例えば、ネットワーク上の遠隔地（ネットワーク統括者の事務所など）からのプリンター制御及び状態観察や、各印刷ジョブ後の次のユーザーのための保証初期環境を提供するためのプリンター構成の自動管理、及びプリンターの負荷量を特徴付け、あるいはトナーカートリッジの交換スケジュールを組むためにネットワークを通してアクセスできる、プリンターログ又は使用統計を含む。

【００２８】このＮＢ設計において重要な要因は、共有メモリ等の両方向インターフェースを介して、ＮＢ１０１からプリンター制御状態にアクセスする機能である。共有メモリ以外に、ＳＣＳＩインターフェース等のインターフェースを使用することもできる。これにより、多数の便利な補助機能のプログラムができるように、プリンター操作情報をＮＢ１０１又は外部ネットワークノードへ送出することができる。印刷画像データ及び制御情報のブロックは、ＮＢ１０１上にあるマイクロプロセッサーによって構成され、共有メモリに記述され、そして、プリンター１０２によって読み込まれる。同様に、プリンター状態情報は、プリンター１０２から共有メモリへ送られ、そこからＮＢプロセッサーによって読み込まれる。

【００２９】図２は、ＮＢ１０１をプリンター１０２にインストールした状態を示す断面図である。図２に示すように、ＮＢ１０１はネットワーク接続の為のフェースプレート１０１ｂを設置した印刷回路ボード１０１ａから構成されており、コネクタ１７０を介してプリンターインターフェースやード１５０に接続されている。プリンターインターフェースカード１５０は、プリンター１０２のプリンターエンジンを直接制御する。印刷データ及びプリンター状態コマンドは、ＮＢ１０１からコネクタ１７０を介して、プリンターインターフェースカード１５０へ入力され、また、プリンター状態情報はプリンターインターフェースカード１５０からやはりコネクター１７０を介して得られる。ＮＢ１０１はこの情報を、フェースプレート１０１ｂのネットワークコネクタを介して、ＬＡＮ１００上で通信する。同時に、プリンター１０２は、従来のシリアルポート１０２ａ及びパラレルポート１０２ｂから、印刷データを受信することもできる。

【００３０】図３は、ＮＢ１０１とプリンター１０２とＬＡＮ１００との電気的接続を示すブロック図である。ＮＢ１０１は、ＬＡＮ１００へはＬＡＮインターフェースを介して、プリンター１０２へはプリンターインターフェースカード１５０を介して直接接続されている。ＮＢ１０１上にはＮＢ１０１を制御するためのマイクロプロセッサー３０１と、マイクロプロセッサー３０１の動作プログラムを格納するためのＲＯＭ３０３と、マイクロプロセッサー３０１がプログラムを実行する上でワークとして用いるためのＲＡＭ３０２と、ＮＢ１０１とプリンタインタフェースカード１５０とが相互にデータをやりとりするための共有メモリ２００があり、内部バスを通じて相互に接続されている。ＮＢ１０１がＳＮＭＰのエージェントとして動作するためのプログラムはＲＯＭ３０３に格納されている。マイクロプロセッサ３０１は、ＲＯＭ３０３に格納されたプログラムに従って動作し、ワークエリアとしてＲＡＭ３０２を用いる。また、プリンターインターフェースカード１５０と相互に通信するためのバッファ領域として共有メモリ２００を用いる。

【００３１】プリンターインタフェースカード１５０上のマイクロプロセッサー１５１はＮＢ１０１とのデータのアクセスを、ＮＢ１０１に設置されている共有メモリ２００を介して行う。プリンターインタフェースカード１５０上のマイクロプロセッサー１５１は、実際に印刷機構を動かすプリンターエンジン１６０とも通信する。

【００３２】一方、ネットワーク管理ソフトウェアが稼動するＰＣ側について、以下で説明する。

【００３３】図５は、ネットワーク管理ソフトウェアが稼動可能なＰＣの構成を示すブロック図である。

【００３４】図５において、５００は、ネットワーク管理ソフトウェアが稼動するＰＣであり、図１における１０３と同等である。ＰＣ５００は、ＲＯＭ５０２もしくはハードディスク（ＨＤ）５１１に記憶された、あるいはフロッピーディスクドライブ（ＦＤ）５１２より供給されるネットワーク管理プログラムを実行するＣＰＵ５０１を備え、システムバス５０４に接続される各デバイスを総括的に制御する。

【００３５】５０３はＲＡＭで、ＣＰＵ５０１の主メモリ、ワークエリア等として機能する。

【００３６】５０５はキーボードコントローラ（ＫＢＣ）で、キーボード（ＫＢ）５０９や不図示のポインティングデバイス等からの指示入力を制御する。５０６はＣＲＴコントローラ（ＣＲＴＣ）で、ＣＲＴディスプレイ（ＣＲＴ）５１０の表示を制御する。５０７はディスクコントローラ（ＤＫＣ）で、ブートプログラム、種々のアプリケーション、編集ファイル、ユーザファイルそしてネットワーク管理プログラム等を記憶するハードディスク（ＨＤ）５１１およびフロッピーディスクコントローラ（ＦＤ）５１２とのアクセスを制御する。５０８はネットワークインタフェースカード（ＮＩＣ）で、ＬＡＮ１００を介して、エージェントあるいはネットワーク機器と双方向にデータをやりとりする。

【００３７】次に、従来例におけるネットワーク管理ソフトウェアの構成について説明する。

【００３８】従来例におけるネットワーク管理装置は、

図５に示したようなネットワーク管理装置を実現可能なＰＣと同様の構成のＰＣ上に実現される。ハードディスク（ＨＤ）５１１には、後述のすべての説明で動作主体となる本願に係るネットワーク管理ソフトウェアのプログラムが格納される。後述のすべての説明において、特に断りのない限り、実行の主体はハード上はＣＰＵ５０１である。一方、ソフトウェア上の制御の主体は、ハードディスク（ＨＤ）５１１に格納されたネットワーク管理ソフトウェアである。本従来例においては、ＯＳは例えば、ウィンドウズ９５（マイクロソフト社製）を想定しているが、これに限るものではない。

【００３９】なお本願に係るネットワーク管理プログラムは、フロッピーディスクやＣＤ－ＲＯＭなどの記憶媒体に格納された形で供給されても良く、その場合には図５に示すフロッピーディスクコントローラ（ＦＤ）５１２または不図示のＣＤ－ＲＯＭドライブなどによって記憶媒体からプログラムが読み取られ、ハードディスク（ＨＤ）５１１にインストールされる。

【００４０】図６は、本従来例に係るネットワーク管理ソフトウェアのモジュール構成図である。このネットワーク管理ソフトウェアは、図５におけるハードディスク５１１に格納されており、ＣＰＵ５０１によって実行される。その際、ＣＰＵ５０１はワークエリアとしてＲＡＭ５０３を使用する。

【００４１】図６において、６０１はデバイスリストモジュールと呼ばれ、ネットワークに接続されたデバイスを一覧にして表示するモジュールである。（一覧表示の様子については、後ほど図１５を用いて説明する。）６０２は全体制御モジュールと呼ばれ、デバイスリストからの指示をもとに、他のモジュールを統括する。６０３はコンフィグレータと呼ばれ、エージェントのネットワーク設定に関する特別な処理を行うモジュールである。６０４は、探索モジュールと呼ばれ、ネットワークに接続されているデバイスを探索するモジュールである。探索モジュール６０４によって探索されたデバイスが、デバイスリスト６０１によって一覧表示される。６０５は、プリントジョブの状況をNetWare API６１６を用いてネットワークサーバから取得するNetWareジョブモジュールである。（なお、NetWare APIについては、例えばNovell社から発行されている"NetWare Programmer*s Guidefor C"等を参照のこと。この書籍はノベル株式会社から購入可能である。）６０６および６０７は後述するデバイス詳細ウィンドウを表示するためのＵＩ(Userlnterface)モジュールであり、詳細情報を表示する対象機種毎にＵＩモジュールが存在する。６０８および６０９は制御モジュールと呼ばれ、詳細情報を取得する対象機種に特有の制御を受け持つモジュールである。ＵＩモジュールと同様に、制御モジュールも詳細情報を表示する対象機種毎に存在する。制御Ａモジュール６０８および制御Ｂモジュール６０９は、ＭＩＢモジュール６１０を用いて管理対象デバイスからＭＩＢデータを取得し、必要に応じてデータの変換を行い、各々対応するＵＩＡモジュール６０６またはＵＩＢモジュール６０７にデータを渡す。

【００４２】さて、ＭＩＢモジュール６１０は、オブジェクト識別子とオブジェクトキーとの変換を行うモジュールである。ここでオブジェクトキーとは、オブジェクト識別子と一対一に対応する３２ビットの整数のことである。オブジェクト識別子は可変長の識別子であり、ネットワーク管理ソフトウェアを実装する上で扱いが面倒なので、本願に係るネットワーク管理ソフトウェアにおいてはオブジェクト識別子と一対一に対応する固定長の識別子を内部的に用いている。ＭＩＢモジュール６１０より上位のモジュールはこのオブジェクトキーを用いてＭＩＢの情報を扱う。これにより、ネットワーク管理ソフトウェアの実装が楽になる。

【００４３】６１１はＳＮＭＰモジュールと呼ばれ、ＳＮＭＰパケットの送信と受信を行う。

【００４４】６１２は共通トランスポートモジュールと呼ばれ、ＳＮＭＰデータを運搬するための下位プロトコルの差を吸収するモジュールである。実際には、動作時にユーザが選択したプロトコルによって、ＩＰＸハンドラ６１３かＵＤＰハンドラ６１４のいずれかがデータを転送する役割を担う。なお、ＵＤＰハンドラは、実装としてWinSock６１７を用いている。（WinSockについては、例えばWindows socket API v1.1の仕様書を参照のこと。このドキュメントは、複数箇所から入手可能であるが、例えばマイクロソフト社製のコンパイラであるVisual C++に同梱されている。）コンフィグレータ６０３が用いる現在のプロトコル６１５というのは、動作時にユーザが選択しているＩＰＸプロトコルかＵＤＰプロトコルのいずれかのことを示す。

【００４５】本従来例で使用する探索モジュール６０４とＭＩＢモジュール６１０との間のインタフェースについて説明する。

【００４６】ＭＩＢモジュール６１０は図９に示すＣ言語のＡＰＩ(Application Program Interface)を上位モジュールに提供する。

【００４７】最初に、上位モジュールはＭＩＢモジュールとの間で、指定したアドレスに対するインタフェース（これをポートと呼ぶ）を開設するために、MIBOpen API９０１呼び出しを行う。ＭＩＢモジュールは、開設されたインタフェースを識別するための識別子（これをポート識別子と呼ぶ）を上位モジュールに返す（MIBOpen API９０１の第一引数portに返される値）。以降上位モジュールはポート識別子を用いてＭＩＢモジュールとのやりとりを行う。

【００４８】ここで指定するアドレスは、動作しているプロトコルのアドレスであり、ＩＰプロトコルの場合はＩＰアドレス、NetWareプロトコルの場合はNetWareアド

レスである。さらにブロードキャストアドレスを指定することもできる。

【００４９】ブロードキャストアドレスを指定してポートをオープンした場合は、ブロードキャストアドレスに応答する複数のデバイスと通信を行うことが可能である。

【００５０】上位モジュールはポートを使用しなくなったときMIBClose API９０４呼び出しを行いポートを閉じる。

【００５１】上位モジュールがＭＩＢオブジェクトの読み出しを行う場合は、MIBReadObjects API９０２呼び出しを行う。MIBReadObjects API９０２呼び出しにはポート識別子、読み出すべきＭＩＢオブジェクトのオブジェクトキーを指定すると共に、ＭＩＢモジュールが読み出したＭＩＢオブジェクトの値を上位へ通知するためのコールバック関数のアドレスを指定する。

【００５２】MIBReadObjects API９０２呼び出しによりＳＮＭＰのＧｅｔ－ｒｅｑｕｅｓｔコマンドが生成され、ネットワーク上に送信される。図８に示したように、このＧｅｔ－ｒｅｑｕｅｓｔコマンドに応答するエージェントを持つデバイスはＧｅｔ－ｒｅｓｐｏｎｓｅコマンドを送信する。

【００５３】上位モジュールがＭＩＢオブジェクトへの書き込みを行う場合は、MIBWriteObjects API９０３呼び出しを行う。MIBWriteObjects API９０３呼び出しにはポート識別子、書き込むＭＩＢオブジェクトのオブジェクトキーとその値を指定すると共に、ＭＩＢモジュールが書き込みの結果を上位へ通知するためのコールバック関数のアドレスを指定する。

【００５４】MIBWriteObjects API９０３呼び出しによりＳＮＭＰのＧｅｔ－ｒｅｑｕｅｓｔコマンドが生成され、ネットワーク上に送信される。図８に示したように、このＧｅｔ－ｒｅｑｕｅｓｔコマンドに応答するエージェントを持つデバイスはＧｅｔ－ｒｅｓｐｏｎｓｅコマンドを送信する。

【００５５】コールバック関数はMIBReadObjects API９０２もしくはMIBWriteObjects API９０３の結果を上位モジュールに通知するためのものである。具体的には、デバイスのアドレスと受信したＧｅｔ－ｒｅｓｐｏｎｓｅコマンドの内容を上位へ通知する。

【００５６】ブロードキャストアドレスを指定してオープンされたポートに対してMIBReadObjects API９０２を呼び出しを行った場合、ネットワーク上に送信されるＧｅｔ－ｒｅｑｕｅｓｔコマンドを運ぶパケット（ＩＰプロトコルの場合はＩＰパケット、NetWareプロトコルの場合はＩＰＸパケット）の宛先アドレスがブロードキャストアドレスになる。従って、このパケットは複数のデバイスで受信されるので、Ｇｅｔ－ｒｅｑｕｅｓｔコマンドには複数のデバイスが応答する。つまりマネージャ側では複数のＧｅｔ－ｒｅｓｐｏｎｓｅコマンドを受信

する。この場合、コールバック関数は、ポート識別子は同じでデバイスのアドレスが異なる複数回の呼び出しが行われる。上位モジュールはアドレス情報を調べることにより、そのコールバックがどのデバイスからのものかを知ることができる。

【００５７】次に具体的なデータの流れを説明する。ＭＩＢモジュール６１０では上位からの要求によりオブジェクトキーからオブジェクトＩＤへの変換等の処理を行いＳＮＭＰモジュール６１１へコマンド送信要求を行う。ＳＮＭＰモジュール６１１はＭＩＢモジュール６１０からの送信要求によりSNMP PDUをＲＡＭ５０３上で組み立て、共通トランスポートモジュール６１２へ送信要求を行う。共通トランスポートモジュール６１２では動作プロトコルによりヘッダの付加等の所定の処理を行い、ＴＣＰ／ＩＰプロトコルであればWinSockモジュール６１７へ、NetWareプロトコルであればNetWare APIモジュール６１６へパケット送信要求を行う。以下ＴＣＰ／ＩＰプロトコルで動作しているものとして説明を行う。WinSockモジュール６１７は送信要求のあったパケットをＩＰパケット化し、ＯＳに対してネットワークへのデータ送信要求を行う。ＯＳはＲＡＭ５０３上のデータをシステムバス５０４を介してＮＩＣ５０８へ書き込む。ＮＩＣ５０８では書き込まれたデータを所定のフレーム化してＬＡＮ１００に送信する。

【００５８】ＬＡＮ１００に接続されているデバイスからのパケットはＮＩＣ５０８で受信される。ＮＩＣ５０８ではパケット受信を割り込みによりＯＳに通知する。ＯＳはＮＩＣ５０８から受信パケットをシステムバス５０４経由で読み出しＲＡＭ５０３に置く。ＯＳでは動作プロトコルもしくは受信したパケットからプロトコルを判断し、ＴＣＰ／ＩＰプロトコルであればWinSockモジュール６１７へ、NetWareプロトコルであればNetWare APIモジュール６１６へパケット受信が通知される。以下ＴＣＰ／ＩＰプロトコルで動作しているものとして説明を行う。WinSockモジュール６１７では、受信パケットが自分宛のものかどうかを受信パケット中のアドレスにより判断する。受信パケットが自分宛のものでは無いときは受信パケットを破棄する。受信パケットが自分宛であった場合は、ＵＰＤハンドラ６１４起動し、共通トランスポートモジュール６１２にパケット受信を通知する。共通トランスポートモジュール６１２ではトランスポートヘッダの除去等の所定の処理を行い、ＳＮＭＰモジュール６１１へパケット受信を通知する。ＳＮＭＰモジュール６１１ではＳＮＭＰヘッダの除去等の所定の処理を行い、ＭＩＢモジュールへＰＤＵ受信を通知する。ＭＩＢモジュール６１０では所定処理を行うと共に受信した情報をMIB APIで規定された形式に変換し上位モジュールのコールバック関数を呼び出すことにより、デバイスからの応答を上位をモジュールへ通知する。

【００５９】なお、以下の説明において、本願に係るネ

ットワーク管理ソフトウェアのことを「NetSpot」と呼称する。

【００６０】NetSpotのインストールに必要なファイルは、通常、フロッピーディスク（ＦＤ）やＣＤ－ＲＯＭなどの物理媒体に記録されて配布されるか、あるいはネットワークを経由して伝送される。ユーザは、これらの手段によりNetSpotのインストールに必要なファイルを入手した後、所定のインストール手順に従ってNetSpotのインストールを開始する。

【００６１】NetSpotのインストール手順は、他の一般的なソフトウェアのインストール手順と同様である。すなわち、ユーザがNetSpotのインストーラをパーソナルコンピュータ（ＰＣ）上で起動すると、その後はインストーラが自動的にインストールを実行する。インストーラは、NetSpotの動作に必要なファイルをＰＣのハードディスクにコピーし、また、必要に応じてユーザから情報を入力してもらいながら、NetSpotの動作に必要なファイルの修正または新規作成なども行う。

【００６２】次に、本従来例におけるネットワーク管理プログラムにおける探索シーケンスについて説明する。

【００６３】図１０は、従来のネットワーク管理プログラムにおける探索シーケンスを示す図である。

【００６４】図１０において、探索モジュール１０３０はネットワーク管理プログラムの探索モジュールを示し、これは図６における６０４と同等である。この探索モジュールは、ネットワーク管理プログラムの他のモジュールと同様に、図１におけるＰＣ１０３上で図５におけるＣＰＵ５０１によって実行される。

【００６５】デバイス１０３１は、ネットワーク上に接続されており、かつＳＮＭＰエージェントが動作しているデバイスでプリンタとして標準的なＭＩＢのみを搭載しているプリンタを示し、例えば図１におけるＮＢ１１８を示す。ここで、標準的なＭＩＢとは、ＲＦＣ１２１３、ＲＦＣ１５１４およびＲＦＣ１７５９に規定されているＭＩＢなどを言う。

【００６６】デバイス１０３２は、ネットワーク上に接続されており、かつＳＮＭＰエージェントが動作しているデバイスで、プリンタとして標準的なＭＩＢおよび企業独自のプライベートＭＩＢを搭載しているプリンタを示し、例えば図１におけるＮＢ１０１を示す。本実施例では、ＮＢ１０１が企業独自のプライベートＭＩＢを実装しているものとして説明を行なう。プライベートＭＩＢは、図４の記号４０８で示されるｃａｎｏｎノード以下のＭＩＢオブジェクトである。

【００６７】上位モジュールから探索開始の指示が出されると、探索モジュールはブロードキャストアドレスを指定してデバイスの状態とデバイス種別を取得するためのＧｅｔリクエストＳＮＭＰパケットをネットワークに送出する（１００１および１００２）。このパケットは、ネットワークに接続されている全てのデバイスに届けられる。

【００６８】ここで、問い合わせるＭＩＢオブジェクトとして、１００１では標準的なＭＩＢのみを、１００２ではプライベートＭＩＢを用いる。異なるＭＩＢオブジェクトを問い合わせるのは、プライベートＭＩＢを搭載しているデバイスと、標準ＭＩＢのみを搭載しているデバイスの両方を見つけたいためである。

【００６９】このＳＮＭＰパケットに対して、ＳＮＭＰエージェントを実装しているネットワークデバイスは、それぞれ応答パケットを送出する（１００３から１００６）。

【００７０】ここで１００３は、標準ＭＩＢを問い合わせるブロードキャストパケット１００１に対するデバイス１０３１の応答であり、デバイス１０３１の状態に応じた値が返る。１００４は、標準ＭＩＢを問い合わせるブロードキャストパケット１００１に対するデバイス１０３２の応答であり、デバイス１０３２の状態に応じた値が返る。１００５は、プライベートＭＩＢを問い合わせるブロードキャストパケット１００２に対するデバイス１０３１の応答であり、デバイス１０３１はプライベートＭＩＢを実装していないため、エラーとしてnoSuchNameが返される。１００６は、プライベートＭＩＢを問い合わせるブロードキャストパケット１００２に対するデバイス１０３２の応答であり、デバイス１０３２の状態に応じた値が返る。

【００７１】探索を開始してから一定の時間が経過すると一時応答タイマー１０２３が満了するので、探索モジュールはさらに詳細な情報を取得すべく、それぞれのデバイスに対してＳＮＭＰパケットを送信する。

【００７２】より具体的には、一時探索タイマー１０２３が満了した時点までに、デバイス１０３１からはプライベートＭＩＢに対してnoSuchNameエラーが返ってきたことがわかっているので、このデバイスをプライベートＭＩＢには対応していないデバイス、すなわち標準ＭＩＢのみに対応しているデバイスとみなして、標準ＭＩＢを用いてより詳細な取得を行なう。この取得およびデバイス１０３１の返答が１００７および１００８である。

【００７３】同様に、デバイス１０３２からは標準ＭＩＢおよびプライベートＭＩＢの両方から正常な応答が返ってきているので、このデバイスをプライベートＭＩＢを実装しているデバイスとみなして、プライベートＭＩＢを用いてより詳細な取得を行なう。

【００７４】この取得およびデバイス１０３２の返答が１００９および１０１０である。

【００７５】さらに時間が経過してデバイス応答タイマー１０２１が満了した時点で、上位モジュールに対してそれまでに探索したデバイスの情報を通知する。

【００７６】さらに時間が経過し、探索間隔タイマー１０２２が満了した時点で、再度探索を開始する。以降今まで説明した動作と全く同一のシーケンスであるので、

説明を省略する。

【００７７】

【発明が解決しようとする課題】しかしながら、上記従来例のようにブロードキャストパケットを送信することは、そのパケットに対してＳＮＭＰに応答できる全てのネットワーク機器が返信パケットを送信することになり、ネットワークのトラフィックを著しく大きくしてしまう。定期的にブロードキャストパケットを送信することはブロードキャストパケットによるトラフィック増大現象が定期的に起きることを意味し、そのような状況はネットワークの負荷および管理の観点から好ましくない。

【００７８】本発明は上記従来例に鑑みてなされたもので、ブロードキャストパケットの使用を最小限に抑え、定期的なネットワークのトラフィックの増大をなくして、ネットワークの負荷を低減させることができるネットワーク管理装置及び方法を提供することを目的とする。

【００７９】

【課題を解決するための手段】上記目的を達成するために、本発明は次のような構成からなる。すなわち、ブロードキャストパケットによってネットワークに接続されているデバイスを検出するためのネットワークデバイス探知手段と、前記ネットワークデバイス探知手段により検出されたデバイスを登録するためのデバイステーブルと、前記デバイステーブルに登録されたデバイスに対して、所定間隔で順次探知シーケンスを遂行する個別デバイス探知手段と、前記個別デバイス探知手段により検出された情報により、前記デバイステーブルを更新するデバイステーブル更新手段とを備える。

【００８０】あるいは、前記デバイステーブルの登録内容の出力を指示された際に、前記デバイステーブルに登録された各デバイスに対して探知シーケンスを遂行し、検出された情報により前記デバイステーブルを更新する手段と、前記デバイステーブルを出力する出力手段とを更に備える。

【００８１】

【発明の実施の形態】以下、添付図面に従って本発明に係る実施形態を詳細に説明する。

【００８２】（第１の実施形態）図１１は、本発明におけるネットワーク管理プログラムとネットワーク上のデバイスが通信する様子を示したシーケンス図である。特に、図１１ではネットワーク管理プログラムがネットワーク上に存在するデバイスを探索する様子を示している。

【００８３】探索モジュール１０３０は本発明におけるネットワーク管理プログラムの探索モジュールを示し、これは図６における６０４と同等である。この探索モジュールは、本発明におけるネットワーク管理プログラムの他のモジュールと同様に、図１におけるＰＣ１０３上で図５におけるＣＰＵ５０１によって実行される。

【００８４】１１０１から１１０７は、探索モジュール１０３０と、ネットワークデバイス１０３１および１０３２との間で通信が行われていることを示す。これらの通信内容については、後述のフローチャートの説明において、供に詳述する。

【００８５】図１２は、探索モジュール６０４が動作の過程において探索するデバイスの情報を記憶するために使用する検出デバイステーブルおよび前回検出テーブルと呼ばれるデータの構造を示す図である。このデータは図５におけるＲＡＭ５０３内に保持されている。

【００８６】検出デバイステーブル１２２０には、検出したデバイスの情報を登録する。デバイスがひとつ検出される毎に、検出デバイステーブルの行が一行追加される。

【００８７】以下、検出デバイステーブルに登録する情報について説明する。

【００８８】ネットワークアドレス１２０１には、検出したデバイスのネットワークアドレスを登録する。ポート１２０２には、ＭＩＢモジュール６１０が提供するＭｉｂＯｐｅｎ関数によって返されるポート値を登録する。

【００８９】デバイスタイプ１２０３には、デバイスの種別を登録する。

【００９０】デバイス状態１２０４には、デバイスが現在どのような状態になっているかを登録する。「活動中」欄１２０５には、デバイスが現在活動中であって、ネットワークを介した通信が可能である場合にはＴＲＵＥを、そうでない場合にはＦＡＬＳＥを登録する。

【００９１】図１２の例においては、検出デバイステーブル１２２０には、現在ふたつのデバイスが登録されている。ひとつは、ネットワークアドレス192.1.16.130のデバイスであり、このデバイスと通信するためのＭＩＢモジュール６１０におけるポート値は０ｘ１２ＦＥである。このデバイスはデバイスタイプがＬＢＰであり、デバイスは使用可能（Ｒｅａｄｙ）であり、活動中であることを示している。また、ネットワークアドレス192.1.16.150のデバイスのポート値は０ｘ２５０１であり、デバイスタイプは複写機（Ｃｏｐｉｅｒ）である。このデバイスでは現在紙詰まり（Ｊａｍ）が発生しており、このデバイスもまた活動中である。

【００９２】このテーブルの使用方法については、後述のフローチャートの説明において、供に詳述する。

【００９３】図１３は、ＰＣ１０３上で動作するネットワークデバイス管理プログラム全体の動作を説明したフローチャートである。

【００９４】プログラムが起動されると、まずステップＳ１３０１においてブロードキャストパケットによるデバイス探索処理を行なう。ここでは、従来例の説明において図１０に示したシーケンスが実行され、ブロードキ

ャストパケットによる探索処理が行われる。検出されたデバイスに関する情報は検出デバイステーブル１２２０に登録される。

【００９５】ブロードキャストパケットによる探索処理が行われるのは、この起動時のみである。

【００９６】次にステップＳ１３０２に進む。ここでは、システムで発生するイベントを待ち、生じたイベントがＳＮＭＰパケットの受信であるか、デバイス応答タイマーの満了であるか、探知ポーリングタイマーの満了であるか、あるいはそれ以外のイベントであるか（全くイベントが生じていない場合も含む）を判断する。

【００９７】次に、発生したイベントに応じて、それぞれ適切なサブルーチンステップＳ１３０３からステップＳ１３０６を呼び出す。

【００９８】それぞれの処理が終了したら、ステップＳ１３０２に合流し、処理を繰り返す。

【００９９】図１４は、探知ポーリング動作を開始する際の動作を示すフローチャートである。

【０１００】このシーケンスが起動される場合は二種類ある。ひとつは、例えばユーザがキーボード５０９あるいは不図示のマウス等の入力機器を用いて例えばメニューを選択するなどの操作によって、明示的にデバイス探知ポーリング動作を開始した場合である。この場合、図１４の処理は、図１３のフローチャートにおいてステップＳ１３０６のその他の処理の一部に相当する。もうひとつの場合は、図１４の処理において起動する探知ポーリングタイマが満了した場合である。この場合には、図１４の処理は、図１３におけるステップＳ１３０５の処理に相当する。

【０１０１】さて、探知ポーリング動作の開始においては、まずステップＳ１４０１において探知ポーリングの対象となるデバイスを示す変数targetに次のデバイスのデバイスのアドレスを代入する。ここで、次のデバイスとは、検出デバイステーブル１２２０において、前回本処理の対象となったデバイスの次のデバイスという意味である。本処理が一度も実行されていない場合には、最初の行のデバイスを示す。検出デバイステーブル１２２０に登録されている全デバイスに対して探知ポーリング動作が終了したかどうかを判断する。なお、検出デバイステーブル１２２０にデバイスがひとつも登録されていない場合には、全デバイス終了と判断する。ステップＳ１４０２において全デバイス終了と判断された場合には、処理を終了する。

【０１０２】一方、まだ終了していないと判断された場合には、ステップＳ１４０３に進み、探知動作を開始する。本ステップの詳細については、後程図１５のフローチャートを用いて説明する。次にステップＳ１４０４に進み、一定時間後に次のデバイスを探知ポーリング動作の対象とするために探知ポーリングタイマを起動する。探知ポーリングタイマが満了したら、図１３におけるステップＳ１３０５の処理、すなわち図１４の処理が開始される。ステップＳ１４０４の処理が終了したら、探知ポーリング動作の処理を終了する。

【０１０３】図１５は、探知動作開始処理を詳しく説明するフローチャートである。本処理は、図１４におけるステップＳ１４０３の処理に相当する。

【０１０４】まずステップＳ１５０１において、探知動作がどこまで進行したかを示す探知フェーズ変数に値１を代入する。この変数は、図１６で説明するパケットの受信処理において使われる。次にステップＳ１５０２に進み、探知動作の対象となるデバイスのデバイス種別を示すＭＩＢオブジェクトに対して、図９で説明したＭｉｂＲｅａｄＯｂｊｅｃｔｓ関数を呼び出すことによってＳＮＭＰのＧｅｔＲｅｑｕｅｓｔを発行してデバイス種別を要求する。次にステップＳ１５０３でデバイス応答タイマーを起動する。デバイスから応答がなく、デバイス応答タイマーが満了すると図１５のステップＳ１３０４の処理が起動される。ステップＳ１５０３でデバイス応答タイマーを起動したら、探知動作開始処理を終了する。

【０１０５】図１６は、パケット受信処理を詳しく説明するフローチャートである。本処理は、図１３におけるステップＳ１３０３に相当する。パケット受信処理においては、まずステップＳ１６０１において、現在探知対象になっているデバイスからのパケット受信であるかどうかを判断する。この判断は、受信したパケットの送信元アドレスが探知対象のデバイスのものであるか、さらにはＳＮＭＰのリクエストＩＤが探知対象のデバイスに送信した値と同じであったかどうかを判断することによって行われる。ステップＳ１６０１において、探知対象のデバイスからのパケット受信であると判断された場合には、ステップＳ１６０２に進み、探知フェーズが１であるかどうかを判断する。探知フェーズが１である場合には、ステップＳ１６０３に進み、デバイスから応答があったので図１５のステップＳ１５０３で起動したデバイス応答タイマーを削除する。

【０１０６】次にステップＳ１６０４に進み、受信したパケットには図１５のステップＳ１５０２で要求したデバイス種別が格納されているはずなので、その値を取り出すとともに、検出デバイステーブル１２２０のデバイスタイプ１２０３の該当する行に取得したデバイスタイプを上書きする。ここで、既存の値を上書きするのは、同じアドレスに対してデバイスが入れ替えられて別のデバイスが設置されている場合があるので、最新の情報を反映するためである。次に、ステップＳ１６０５では、探知シーケンスの進み具合を示す探知フェーズ変数に値２を代入する。次にステップＳ１６０６に進み、ＭｉｂＲｅａｄＯｂｊｅｃｔｓ関数を用いてデバイスに対してデバイス状態を示すＭＩＢオブジェクトの値を要求する。

【０１０７】さて、ステップＳ１６０２において探知フェーズが１ではないと判断された場合、探知フェーズはデバイス状態を要求したフェーズ２であるのでステップＳ１６０７に進み、受信パケットの中からデバイス状態を示す値を取り出して、検出デバイステーブル１２２０のデバイス状態１２０４の該当する行に取り出した値を格納する。次にステップＳ１６０８に進み、探知対象のデバイスから必要な値全てが取り出すことが出来てデバイスが活動中であることが確認できたので、デバイステーブル１２２０の活動中欄１２０５の該当する行に活動中を示す値であるＴＲＵＥを書き込む。

【０１０８】最後に図１３のステップＳ１３０４において、デバイス応答タイマーが満了した場合の処理について説明する。デバイス応答タイマーが満了するのは、図１５のステップＳ１５０２におけるデバイスへのデバイス種別値の要求に対してデバイスが応答を返さなかった場合である。このような場合、デバイスは電源が切られているか、あるいは既にネットワークから切り離された状態であると考えられるため、図１２の検出デバイステーブル１２２０の活動中欄１２０５の該当行に活動中ではないことを示す値ＦＡＬＳＥを登録するとともに、ＣＲＴＣ５０６を操作してＣＲＴにその旨を表示し、ユーザに通知する。

【０１０９】ユーザへの表示の様子を図１８に示す。ウィンドウ１８０１は、検出デバイステーブルに登録されているデバイスが表示されており、それらのうち探知シーケンスによって活動中ではないと判断されたデバイスは、×印１８０３がつけられている。

【０１１０】以上説明したように、本実施例によれば、定期的にブロードキャストパケットを送信することによりデバイステーブルを更新する代わりに、デバイステーブルに載っているデバイスに対して順番にその状態を検出するためのパケットを送信して、デバイスが反応するかどうかだけを調べることができる。これにより比較的少ないネットワークトラフィックでデバイステーブルを更新することができる。

【０１１１】（第２の実施形態）第１の実施例によると、探索におけるブロードキャストパケットの送信が最初の一回だけに限定され、その後はデバイス毎に比較的長い時間間隔で、しかも一度にひとつのデバイスに対して探知シーケンスが起動されるため、ネットワークのトラフィックが一時的に増大するようなことはない。

【０１１２】しかしながら、ひとつあたりのデバイスで考えた場合、探知シーケンスが起動されてから、次同じデバイスに対して探知シーケンスが実行されるまでの間隔は、ブロードキャストを用いていた従来例に比較して長くなっている。このような状況下では、古いネットワークプリンタをネットワークから切り離して、代わりに新しいプリンタを設置したような場合には、検出デバイステーブルに登録されている機種情報が実際に設置されているデバイスと異なる可能性が高い。

【０１１３】そのような場合にデバイス詳細ウィンドウ１８０２を開いても、デバイス種別が異なるために、正しい情報が得られない。

【０１１４】そこで本発明の第２の実施例では、デバイス詳細ウィンドウを開く前には必ず探知シーケンスを実行することにより、検出デバイステーブルの情報を最新の情報に更新するとともに必ず正しくデバイス詳細ウィンドウを開けるようにしたものである。

【０１１５】以下、フローチャートを用いて説明する。

【０１１６】図１７は、本発明の第２の実施例の動作を説明するフローチャートである。第２の実施例において、このフローチャート以外の構成については基本的には第１の実施例と同じである。

【０１１７】図１７で示されるデバイス詳細ウィンドウを開く手順は、ユーザがメニューからデバイス詳細ウィンドウを開く操作を行なった時に起動され、第一の実施例における図１３のステップＳ１３０６のその他の処理の一部となる。

【０１１８】本手順では、まずステップＳ１７０１においてユーザがデバイス詳細ウィンドウを開こうとしたデバイスに対して探知シーケンスを起動する。このシーケンスは、第１の実施例においてデバイス一台あたりの情報を取得するまでに相当する。次にステップＳ１７０２において探知された情報と検出デバイステーブル１２２０の情報とを比較して、それらが同じ内容であるかどうかを調べる。もし同じ情報であればステップＳ１７０３に進み、デバイス詳細ウィンドウ１８０２を開き、デバイスから詳細な情報を取得してその内容をウィンドウ１８０２に表示する。一方ステップＳ１７０２において異なる情報であると判断された場合にはステップＳ１７０４に進み、検出デバイステーブル１２２０の情報を更新する。次にステップＳ１７０５に進み、例えばポップアップウィンドウを表示するなどの方法によってユーザにデバイスが入れ替えられている旨を通知する。次にステップＳ１７０３の処理に合流する。

【０１１９】以上のように、本実施例によれば、デバイス詳細ウィンドウを開くときに、デバイスとネットワークボードの種別を確かめる探知シーケンスを必ず実行することができる。これにより、デバイスの探索の頻度が下がっても、デバイス参照ウインドウには、それが開かれた際に最新の状態を確実に表示することができる。このため、探知間隔よりも短いタイミングでデバイスあるいはネットワークボードが異なる種別のものに入れ替えられていたときであっても正しくデバイス詳細ウィンドウを開くことができる。

【０１２０】上記で説明した本発明に係るネットワークデバイス制御プログラムは、外部からインストールされるプログラムによって、ＰＣ５００によって遂行されても良い。その場合、そのプログラムはＣＤ－ＲＯＭやフ

ラッシュメモリやフロッピーディスクなどの記憶媒体により、あるいは電子メールやパソコン通信などのネットワークを介して、外部の記憶媒体からプログラムを含む情報群をＰＣ５００上にロードすることにより、ＰＣ５００に供給される場合でも本発明は適用されるものである。

【０１２１】図１９は、記憶媒体の一例であるＣＤ－ＲＯＭのメモリマップを示す図である。９９９９はディレクトリ情報を記憶してある領域で、以降のインストールプログラムを記憶してある領域９９９８およびネットワークデバイス制御プログラムを記憶してある領域９９９７の位置を示している。９９９８は、インストールプログラムを記憶してある領域である。９９９７は、ネットワークデバイス制御プログラムを記憶してある領域である。本発明のネットワーク制御プログラムがＰＣ５００にインストールされる際には、まずインストールプログラムを記憶してある領域９９９８に記憶されているインストールプログラムがシステムにロードされ、ＣＰＵ５０１によって実行される。次に、ＣＰＵ５０１によって実行されるインストールプログラムが、ネットワークデバイス制御プログラムを記憶してある領域９９９７からネットワークデバイス制御プログラムを読み出して、ハードディスク５１１に格納する。

【０１２２】なお、本発明は、複数の機器（例えばホストコンピュータ、インタフェース機器、リーダなど）から構成されるシステムあるいは統合装置に適用しても、ひとつの機器からなる装置に適用してもよい。

【０１２３】また、前述した実施形態の機能を実現するソフトウェアのプログラムコードを記録した記憶媒体を、システムあるいは装置に供給し、そのシステムあるいは装置のコンピュータ（またはＣＰＵやＭＰＵ）が記憶媒体に格納されたプログラムコードを読み出し実行することによっても、本発明の目的が達成されることは言うまでもない。

【０１２４】この場合、記憶媒体から読み出されたプログラムコード自体が本発明の新規な機能を実現することになり、そのプログラムコードを記憶した記憶媒体は本発明を構成することになる。

【０１２５】プログラムコードを供給するための記憶媒体としては、例えば、フロッピーディスク、ハードディスク、光ディスク、光磁気ディスク、ＣＤ－ＲＯＭ、ＣＤ－Ｒ、磁気テープ、不揮発性のメモリカード、ＲＯＭなどを用いることができる。

【０１２６】また、コンピュータが読み出したプログラムコードを実行することによって、前述した実施形態の機能が実現される他、そのプログラムコードの指示に基づき、コンピュータ上で稼動しているＯＳなどが実際の処理の一部または全部を行い、その処理によっても前述した実施形態の機能が実現され得る。

【０１２７】さらに、記憶媒体から読み出されたプログラムコードが、コンピュータに挿入された機能拡張ボードやコンピュータに接続された機能拡張ユニットに備わるメモリに書き込まれた後、そのプログラムコードの指示に基づき、その機能拡張ボードや機能拡張ユニットに備わるＣＰＵなどが実際の処理の一部または全部を行い、その処理によっても前述した実施形態の機能が実現され得る。

【０１２８】なお、本発明は、前述した実施形態の機能を実現するソフトウェアのプログラムコードを記録した記憶媒体から、そのプログラムをパソコン通信など通信ラインを介して要求者にそのプログラムを配信する場合にも適用できることは言うまでもない。

【０１２９】

【発明の効果】以上説明したように、本発明の第一の実施例によれば、定期的にブロードキャストパケットを送信することによりデバイステーブルを更新する代わりに、デバイステーブルに載っているデバイスに順番にパケットを送信して、デバイスが反応するかどうかだけを調べることができる。これにより比較的少ないネットワークトラフィックでデバイステーブルを更新することができる。

【０１３０】さらに本発明の第二の実施例によれば、デバイス詳細ウィンドウを開くときに、デバイスとネットワークボードの種別を確かめる探知シーケンスを必ず実行することができる。これにより、探知間隔よりも短いタイミングでデバイスあるいはネットワークボードが異なる種別のものに入れ替えられていたときであっても正しくデバイス詳細ウィンドウを開くことができる。

【図面の簡単な説明】

【図１】プリンタをネットワークに接続するたあのネットワークボードを、開放型アーキテクチャを持つプリンターへつなげた場合を示す図である。

【図２】エージェントを実装したネットワークボードをプリンタに接続する実施形態を示す断面図である。

【図３】ネットワークボードとプリンターとＬＡＮとの電気的接続を示すブロック図である。

【図４】ＭＩＢの構造を示す概念図である。

【図５】ネットワーク管理ソフトウェアが稼動可能なＰＣの構成を示すブロック図である。

【図６】ネットワーク管理ソフトウェアのモジュール構成図である。

【図７】ＳＮＭＰメッセージのフォーマットを示す図である。

【図８】マネージャとエージェント間でのＳＮＭＰコマンドのやりとりを示す図である。

【図９】ＭＩＢモジュールＡＰＩを示す図である。

【図１０】従来例における探索モジュールとデバイス間の通信シーケンスを示す図である。

【図１１】本発明の実施例における探索モジュールとデバイス間の通信シーケンスを示す図である。

【図１２】検出デバイステーブルの構造を示す図である。

【図１３】ネットワークデバイス管理プログラム全体の動作を説明したフローチャートである。

【図１４】探知ポーリング動作を開始する際の動作を示すフローチャートである。

【図１５】探知動作開始処理を詳しく説明するフローチャートである。

【図１６】パケット受信処理を詳しく説明するフローチャートである。

【図１７】本発明の第２の実施例の動作を説明するフロー

ーチャートである。

【図１８】ユーザへの画面表示の方法を一例を示す図である。

【図１９】本発明のネットワーク管理ソフトウェアの記憶媒体におけるメモリマップを示す図である。

【符号の説明】

Ｓ１３０１　ブロードキャストパケット送信手段

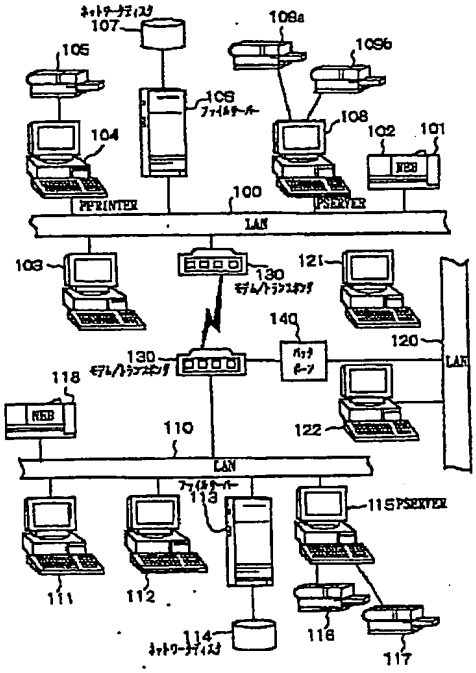
Ｓ１２２０　検出デバイステーブル

Ｓ１４０４　探知ポーリングタイマー起動手段

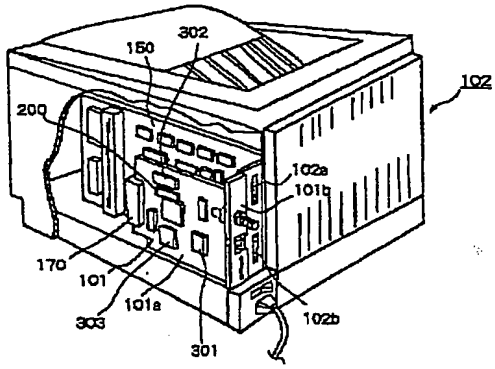
Ｓ１４０３　探知シーケンス開始手段

Ｓ１６０８　検出デバイステーブル更新手段

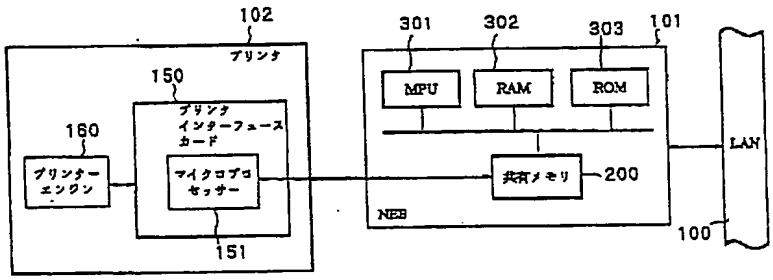
【図１】



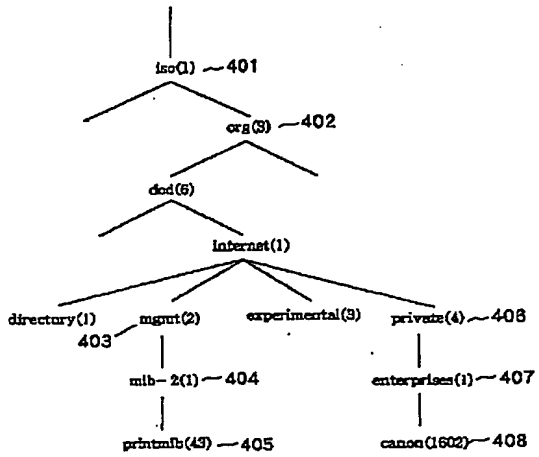
【図２】



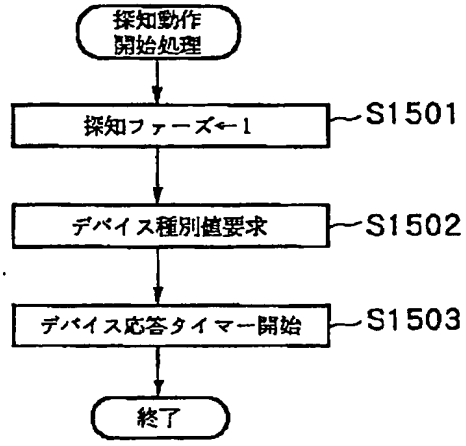
【図３】



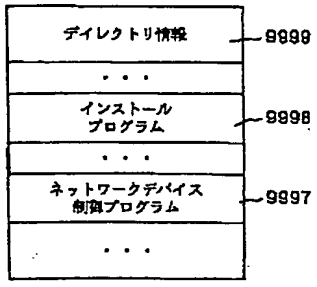
【図４】


iso(1) 401
org(3) 402
dod(6)
internet(1)
directory(1)
mgmt(2)
403
experimental(3)
private(4) 408
mib-2(1) 404
enterprises(1) 407
printmib(43) 405
canon(1602) 408

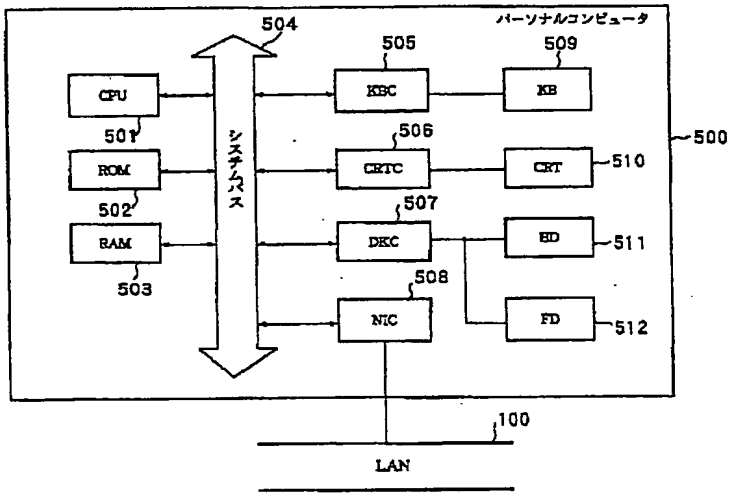
【図１５】


採知動作
開始処理
採知ファーズ←1
S1501
デバイス種別値要求
S1502
デバイス応答タイマー開始
S1503
終了

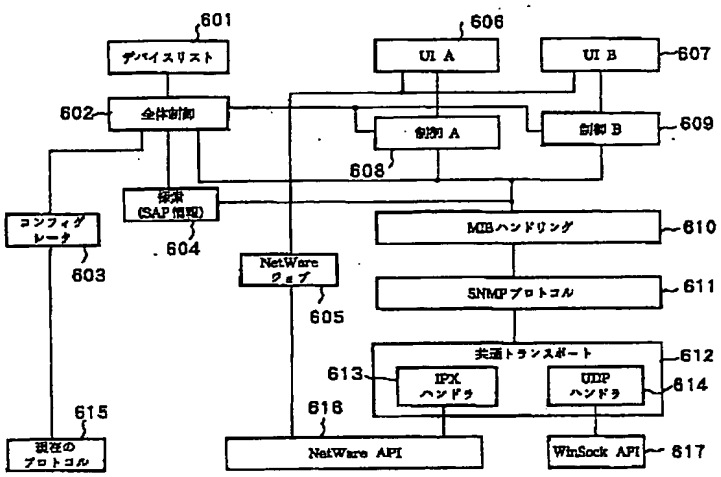
【図１９】


ディレクトリ情報
9999
インストール
プログラム
9998
ネットワークデバイス
制御プログラム
9997

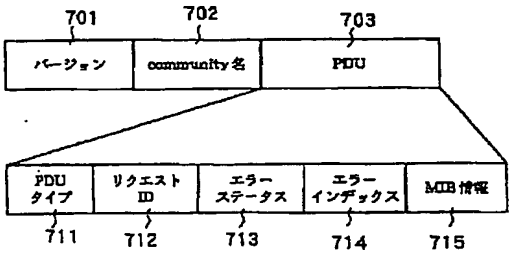
【図５】


パーソナルコンピュータ
504
505
509
CPU
KBC
KB
501
506
ROM
CRTC
CRT
510
500
502
507
RAM
DKC
HD
511
503
508
NIC
FD
512
100
LAN

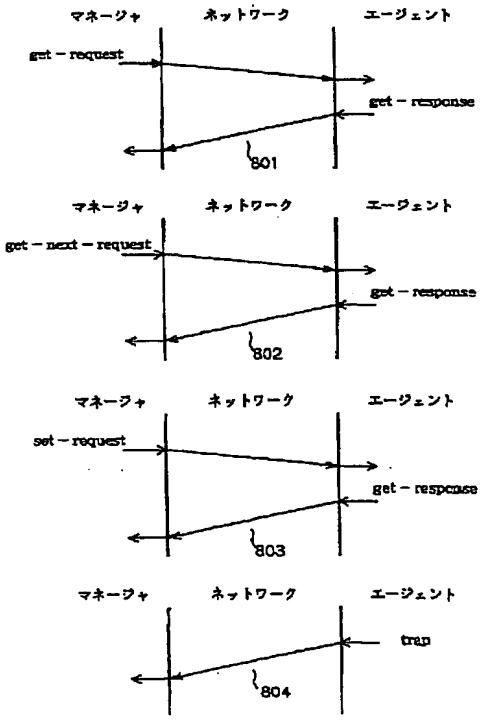
【図６】


601
デバイスリスト
606
UI A
UI B
607
602
全体制御
制御 A
制御 B
609
608
(SAP情報)
604
コンフィグ
レータ
603
MIBハンドリング
610
NetWare
クライアント
605
SNMPプロトコル
611
共通トランスポート
612
613
IPX
ハンドラ
UDP
ハンドラ
614
615
616
現在の
プロトコル
NetWare API
WinSock API
617

【図７】


701
702
703
バージョン
community名
PDU
PDU
タイプ
リクエスト
ID
エラー
ステータス
エラー
インデックス
MIB情報
711
712
713
714
715

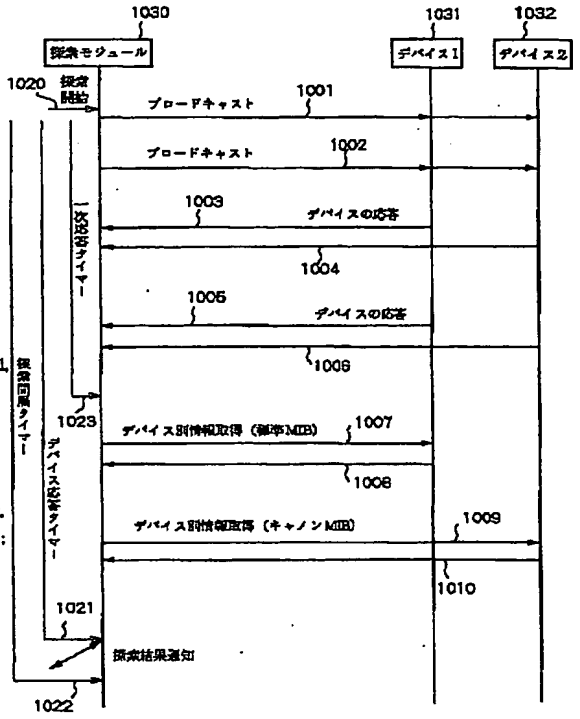
【図８】


マネージャ
ネットワーク
エージェント
get－request
get－response
801
マネージャ
ネットワーク
エージェント
get－next－request
get－response
802
マネージャ
ネットワーク
エージェント
set－request
get－response
803
マネージャ
ネットワーク
エージェント
trap
804

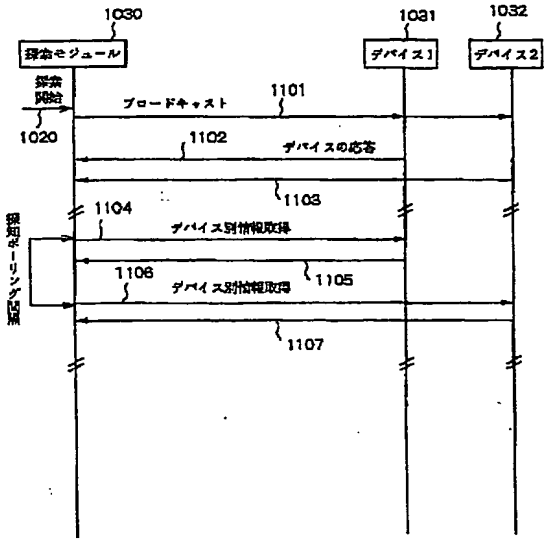
【図９】

```
BOOL MIBOpen (int *port, ADDR  ADDR *addr ) ;                    901
BOOL MIBReadObjects (int  port, int count, MIBOBJ *obj,          902
                     CALLBACK respproc ) ;
BOOL MIBWriteObjects (int  port, int count, MIBOBJVAL *objval,   903
                     CALLBACK respproc ) ;
BOOL MIBClose( int port ) ;                                      904
typedef VOID (*CALLBACK) (int  port, ADDR *addr, INT result,     905
                     int count, MIBOBJVAL *objval ) ;
```

【図１０】



【図１１】

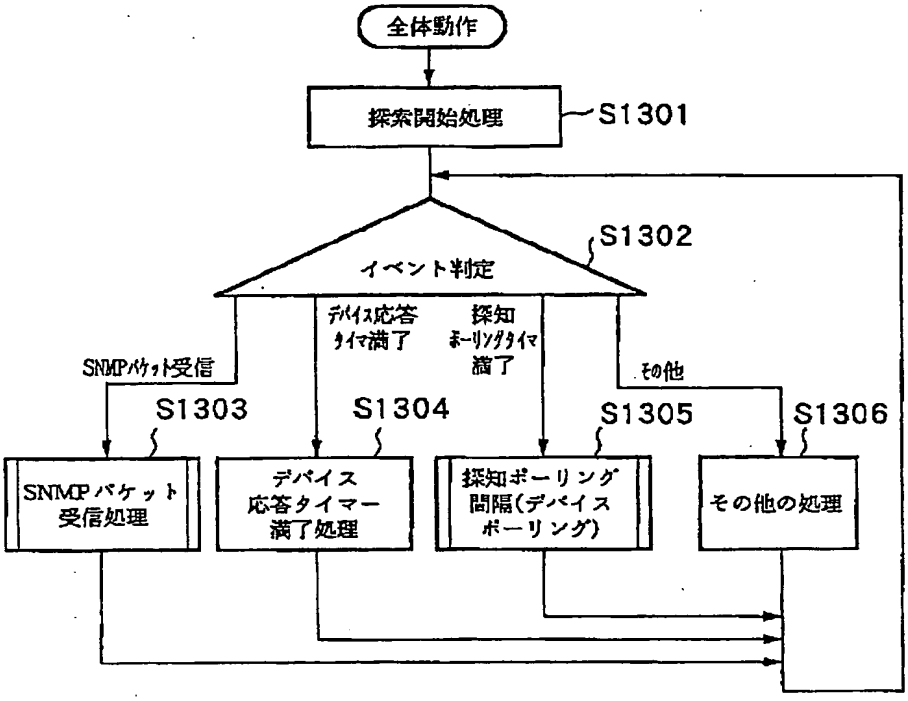


【図１２】

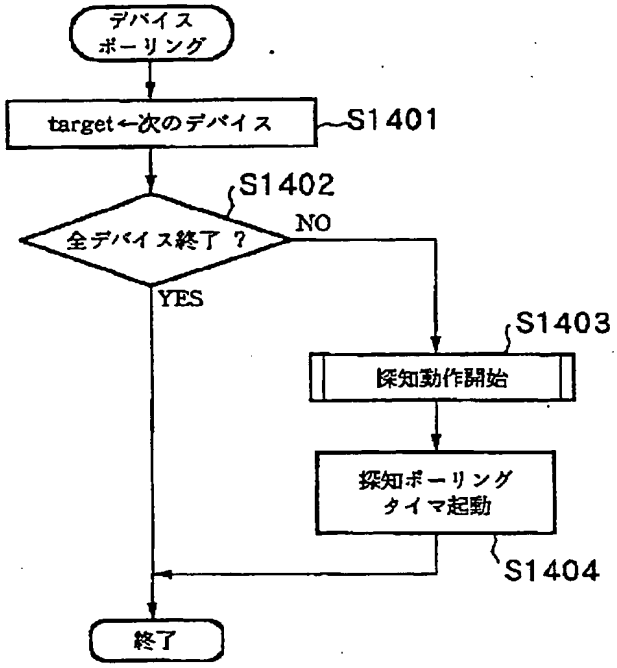
検出デバイステーブル1220

1201	1202	1203	1204	1205
ネットワークアドレス	ポート	デバイスタイプ	デバイス名称	活動中
192.168.16.130	0x12FE	LBP	Ready	TRUE
192.168.16.150	0x2501	Copier	Jam	TRUE
…	…	…	…	…

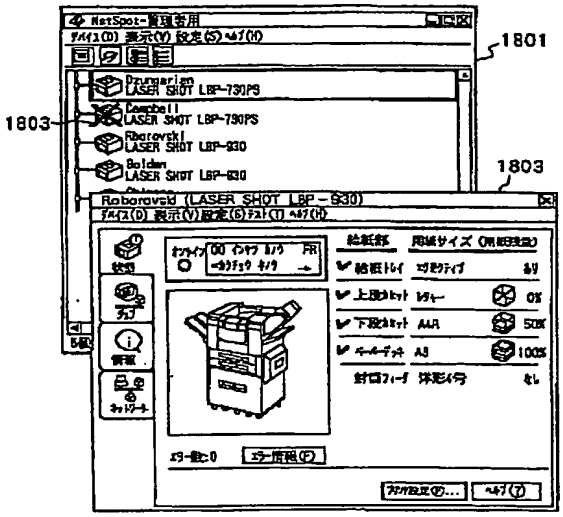
【図１３】


全体動作
探索開始処理
S1301
S1302
イベント判定
SNMPパケット受信
デバイス応答タイマ満了
探知ポーリングタイマ満了
その他
S1303
S1304
S1305
S1306
SNMPパケット受信処理
デバイス応答タイマー満了処理
探知ポーリング間隔(デバイスポーリング)
その他の処理

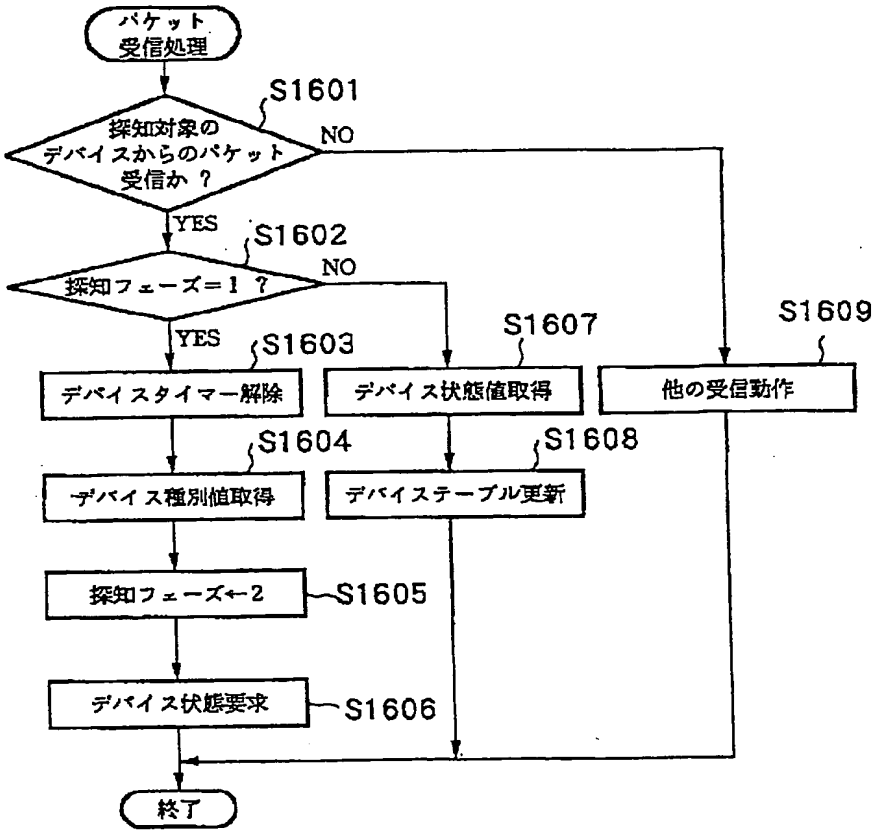
【図１４】


デバイス
ポーリング
target←次のデバイス
S1401
S1402
全デバイス終了 ?
NO
YES
S1403
探知動作開始
探知ポーリング
タイマ起動
S1404
終了

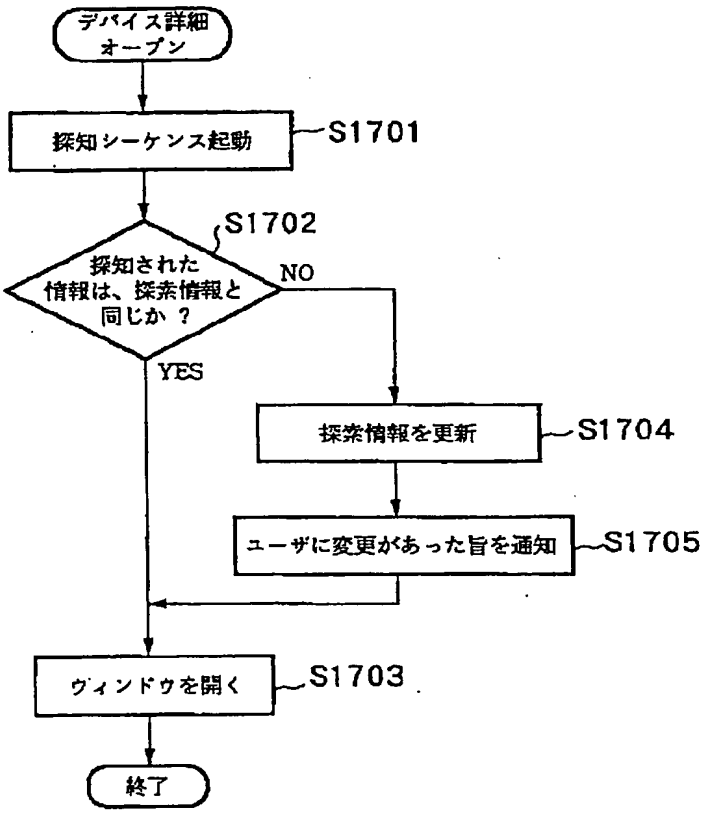
【図１８】


1801
1803
1803
LASER SHOT LBP-730PS
LASER SHOT LBP-930
LASER SHOT LBP-930
Roborovski (LASER SHOT LBP-930)

【図１６】


パケット
受信処理
S1601
探知対象の
デバイスからのパケット
受信か ？
NO
YES
S1602
探知フェーズ＝１ ？
NO
YES
S1603
S1607
S1609
デバイスタイマー解除
デバイス状態値取得
他の受信動作
S1604
S1608
デバイス種別値取得
デバイステーブル更新
探知フェーズ←2
S1605
デバイス状態要求
S1606
終了

【図１７】


デバイス詳細
オープン
探知シーケンス起動
S1701
S1702
探知された
情報は、探索情報と
同じか ？
NO
YES
探索情報を更新
S1704
ユーザに変更があった旨を通知
S1705
ウィンドウを開く
S1703
終了